# General Disclaimer



- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.

- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.

- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.

- This document is paginated as submitted by the original source.

- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

X-641-70-39
PREPRINT

NASA TM X- 63819

# ASP

## ARTIFICIAL SCIENTIFIC PROGRAMMING

ROBERT BAXTER
EDWARD SULLIVAN

FEBRUARY 1970



GSFC
GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND

08

# ASP

# ARTIFICIAL SCIENTIFIC PROGRAMMING

by

Robert Baxter
and
Edward Sullivan

# INTRODUCTION

Artificial Scientific Programming (ASP) is an application language designed to facilitate the computer solution of a large class of scientifically oriented problems. The salient feature of an application language is the minimum effort required to program a computer solution.

The ASP language is implemented on an IBM 360/91 with an MVT job scheduler. Input is accepted from cards, a remote input terminal system RITS, or a question-answer session using the 2250 display unit. The free form statements are translated into a Fortran IV program, which is optionally compiled and executed. Programs generated by ASP are readable and flexible enough to make basic alterations a simple process, because general numerical algorithms frequently require modification to solve a difficult problem.

In this report, the ASP language elements necessary to solve a system of first order ordinary differential equations are defined. A fifth order floating point version of Nordsieck's method[1] is used to obtain solutions of first order initial value problems. Nordsieck incorporates iterative self starting, and

automatic revision of the step length into an efficient general purpose method. The step length is chosen such that a relative measure of the truncation error per integration step is less than a specified bound. To ensure the stability of the differential equations, it is necessary to compute the spectral radius of the Jacobian matrix[2]. Consequently, the assurance of stability is left as a user supplied constraint on the step length.

## ASP SYNTAX

The following conventions are adhered to in the description of the ASP language:

1. Statements are order independent unless explicitly stated otherwise, while expressions within a particular statement field are evaluated in the order they appear. Blanks are ignored except for headings.

2. Variable names, arithmetic expressions, and functions should be written following the normal Fortran conventions, with the restriction that names be no more than four characters.

3. A lower case e represents a Fortran arithmetic expression, c a floating point constant, k an integer constant, and v a variable name.

The first statement must be one of the three listed below:

SOLVE; ODE:

PROGRAM; DECK; ODE:

PROGRAM; RITS; ODE:

For each of the above a Fortran program is generated which will solve a system of first order ordinary differential equations. If the first form is used the resulting Fortran program will be compiled and executed. Two and three will have the program punched on cards, or written on a disk so that it may be moved into a RITS

library. The differential equations to be solved must immediately follow the SOLVE or PROGRAM request and are expressed as either

$Y(1,1)=e_1$; $Y(1,2)=e_2$; --- $Y(1,n)=e_n$:

or

$F(1)=e_1$; $F(2)=e_2$; --- $F(n)=e_n$:

where Y(1,i) represents the first derivative of $Y_i$ with respect to T. The generated program is written to solve a system of equations with the standard nomenclature

$$\frac{dY}{dT} = F(T,Y)$$

where Y and F are vectors. If it is more convenient to type in the equations with names other than the standard, the following designation must appear

VARIABLES: $v_1$; $v_2$; $v_3$:

For example, the system represented by

$$\frac{dP}{dR} = G(R,P)$$

may have the ASP definition

$P(1,1)=e_1$; $P(1,2)=e_2$; --- $P(1,n)=e_n$:

or

$G(1)=e_1$; $G(2)=e_2$; --- $G(n)=e_n$:

VARIABLES: R; P; G:

If "computational" equations facilitate the writing of the differential equations, they are specified as

EQUATIONS: $v_1=e_1$; $v_2=e_2$; --- $v_n=e_n$:

Summations are indicated by the A3P function

SIGMA(v=$k_1$,$k_2$; e)

To illustrate the way the EQUATION statement and the summation function are used consider the system of equations

$$Y_i = Y_i/(Y_1+Y_2+Y_3)$$

This system may be written

```
Y(1,1)=Y(1)/R; Y(1,2)=Y(2)/R; Y(1,3)=Y(3)/R;
EQUATIONS: S=SIGMA(I=1,3; Y(I)); R=SQRT(S):
```

SIGMA may only appear in assignment statements of the form $v=e_1+e_2$*SIGMA(---), and its use is restricted to the EQUATION and DYNAMIC statements. The summand in the SIGMA function may contain nested sums such as the double sum

```
S2=SIGMA(I=1,10; A(I)*SIGMA(J=1,5; B(I,J)))
```

The size of arrays that appear in the program other than Y and F is transmitted by

ARRAYS: $v_1(\underline{k}_1)$; $v_2(\underline{k}_2)$; --- $v_n(\underline{k}_n)$:

where $\underline{k}$ denotes a vector of seven or less elements. A convenient way to initialize arrays is to use

DATA: $v_1$, $v_2$, --- $v_n$; $k_1$*$c_1$, $k_2$*$c_2$, --- $k_n$*$c_n$:

or

```
TABULAR: list; format:
```

The construction of these statements is best exemplified

by their ASP Fortran translations

DATA $v_1$, $v_2$, --- $v_n$ / $k_1$*$c_1$, $k_2$*$c_2$, --- $k_n$*$c_n$ /

and

READ (5,xx) list

xx FORMAT (format)

One procedure to initialize the matrices

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix} \qquad B = \begin{bmatrix} 2 & 3 & 4 \\ 3 & 4 & 5 \\ 4 & 5 & 6 \end{bmatrix}$$

is to specify

ARRAYS: A(3,3); B(3,3):

DATA A: 3*1.0, 3*2.0, 3*3.0:

TABULAR: ((B(I,J),I=1,3),J=1,3); 3F5.1:

The data cards associated with each TABULAR statement follow the ASP program, and are arranged in the order requested. Statements used to represent parameters and Fortran arithmetic functions are of the form

PARAMETERS: $v_1$=$c_1$; $v_2$=$c_2$; --- $v_n$=$c_n$:

FUNCTIONS: $v_1(\underline{v}_1)$=$e_1$; $v_2(\underline{v}_2)$=$e_2$; --- $v_n(\underline{v}_n)$=$e_n$:

The interval of integration and the print increment are defined in

INTERVAL: $c_1$; $c_2$: PRINT: $c_3$(default=1.0):

Backward integration is implicitly defined by $c_1 > c_2$.

$c_3$ must be positive, and normally the solution will be

printed for T=$c_1$, $c_1+c_3$, --- $c_1+kc_3$, $c_2$. Initial conditions are given with

INITIAL: Y(1)=$e_1$; Y(2)=$e_2$; --- Y(n)=$e_n$:

Frequently, it is necessary to have functions dependent on the integrated values printed out along with the solution. This is accomplished by including

DYNAMIC: $v_1$=$e_1$; $v_2$=$e_2$ --- $v_n$=$e_n$:

Another useful calculation associated with solutions of differential equations is the evaluation of integrals. Integration is specified in general by

INTEGRAL(v; e)

The limits of integration are assumed to be those in the INTERVAL statement. For example to evaluate

$$A = \int \exp(-T)\ Y_1\ dT$$

it is necessary to include

INTEGRAL(A;EXP(-T)*Y(1)):

If the differential equations contain indeterminate forms over the interval of integration the limiting values are defined as

SINGULAR: T=c; F(1)=$e_1$; F(2)=$e_2$; --- F(n)=$e_n$:

The default value for a bound on the relative truncation error per step is $10^{-5}$. This may be changed by stating

PRECISION: k:

If $k \leq 5$ a single precision program is written, and double precision if $k>5$. When defining a double precision

program, it is necessary to use the corresponding double precision Fortran functions if any appear in the ASP expressions. If a fixed step length and no automatic truncation error testing or step length modification is desired

STEP: $c_1$; $c_2$:

must be included. The step length will have the value $c_1$. The constant $c_2$ is optional and if inserted signifies that truncation error testing is to be resumed when the independent variable of integration is greater than $c_2$. A heading to be printed on each page of output is denoted

HEADING: 80 or less alphanumeric characters:

Comments may be inserted anywhere in the program and are enclosed by double quotes

" any appropriate comment "

To have a condensed version of the output written on a disk so that a listing on the remote terminal is possible the statement

OUTPUT: TERMINAL:

must be included.

## EXAMPLES

For the first example a restricted three body problem[3] (earth-moon-spaceship) is considered

$$\frac{d^2x}{dt^2} = x + 2\frac{dy}{dt} - a'(x + g) - a(x - g')$$

$$\frac{d^2y}{dt^2} = y - 2\frac{dx}{dt} - a'y - ay$$

with

$$g = 1/82.45,\ g' = 1 - g$$

$$a = g / ((x - g')^2 + y^2),\ a' = g' / ((x + g)^2 + y^2)$$

and initial values

$$t=0,\ x_0=1.2,\ x_0'=0,\ y_0=0,\ y_0'=-1.04935750983.$$

The solution x(t), y(t) is a closed orbit with period t=6.192169331396. After a change of variable to rewrite the equations as a first order system an ASP program for this example is

```
SOLVE; ODE:
F(1)=U(3); F(2)=U(4);
F(3)=U(1)+2.0*U(4)-AP*(U(1)+g)-A*(U(1)-GP);
F(4)=U(2)-2.0*U(3)-U(2)*(AP+A):
VARIABLES: T; U; F:
EQUATIONS: A=G/DSQRT((U(1)-GP)**2+U(2)**2)**3;
AP=GP/DSQRT((U(1)+G)**2+U(2)**2)**3:
```

```
PARAMETERS: G=1.2128562765312D-2;
GP=9.8787143723469D-1:
INITIAL: U(1)=1.2D+0; U(2)=0.0D+0; U(3)=0.0D+0;
U(4)=-1.04935750983:
PRECISION: 9: PRINT: 6.192169331396D-1:
INTERVAL: 0.0D+0; 6.192169331396:
HEADING: RESTRICTED 3-BODY PROBLEM::
```

Two successive colons indicate the end of the ASP program. The resulting solution is listed in Appendix B.

The next example is a solution of the regular Coulomb function[4] which satisfies the differential equation

$$\frac{d^2w}{d\rho^2} + \left(1 - 2\frac{\eta}{\rho} - \frac{L(L+1)}{\rho^2}\right)w = 0$$

for $L=0, \eta=1/2$. An ASP program to solve this equation is

```
SOLVE; ODE:
  W(1,1)=W(2);
  W(1,2)=(-1.0+2.0*ETA/RHO-EL*(EL+1)/RHO**2)*W(1):
VARIABLES: RHO; W; F:
PARAMETERS: EL=0.0; ETA=0.5; CL=3.766858E-1:
INITIAL: W(1)=0.0; W(2)=CL:
SINGULAR: RHO=0.0; W(1,1)=CL; W(1,2)=CL:
HEADING: COULOMB FUNCTION:
INTERVAL: 0.0; 10.0: PRINT: 1.0::
```

The solution from this program is also in Appendix B.

## APPENDIX A

## SUMMARY OF THE LANGUAGE ELEMENTS

ARRAYS: $v_1(\underline{k}_1)$; $v_2(\underline{k}_2)$; --- $v_n(\underline{k}_n)$:

DYNAMIC: $v_1=e_1$; $v_2=e_2$; --- $v_n=e_n$:

EQUATIONS: $v_1=e_1$; $v_2=e_2$; --- $v_n=e_n$:

FUNCTIONS: $v_1(\underline{v}_1)=e_1$; $v_2(\underline{v}_2)=e_2$; --- $v_n(\underline{v}_n)=e_n$:

HEADING: 80 or less alphanumeric characters :

INITIAL: $Y(1)=e_1$; $Y(2)=e_2$; --- $Y(n)=e_n$:

INTEGRAL(v; e):

INTERVAL: $c_1$; $c_2$:

OUTPUT: TERMINAL:

PARAMETERS: $v_1=c_1$; $v_2=c_2$; --- $v_n=c_n$:

PRECISION: k:

PRINT: c:

SIGMA(v=$k_1$,$k_2$; e)

STEP: $c_1$; $c_2$:

TABULAR: list; format:

DATA: $v_1$, $v_2$, --- $v_n$; $k_1*c_1$, $k_2*c_2$, --- $k_n*c_n$:

VARIABLES: $v_1$; $v_2$; $v_3$:

" any comment "

## APPENDIX B

## PROGRAM OUTPUT

```
ASP (ODE-A) PROGRAM OUTPUT      RESTRICTED 3 BODY PROBLEM

PARAMETERS
G= 1.212856276531D-02      GP= 9.8787143723469D-01

 T                      U(I), I=1,NE OR U(I)/F(I), I=1,NE
0.0                     1.2000000000000D 00  0.0
6.1921693313960D-01  9.0356308532460D-01 -5.1736905428703D-01
1.2384338662792D 00  3.1698908458685D-01 -3.9249027884245D-01
1.8576507994188D 00 -4.3088791432671D-01 -5.8767318174133D-01
2.4768677325584D 00 -9.9989012077892D-01 -5.5615662740507D-01
3.0960846656980D 00 -1.2624543543930D 00  7.6927307460315D-09
3.7153015988376D 00 -9.9989013924600D-01  5.5615665735015D-01
4.3345185319772D 00 -4.3088793941594D-01  5.8767326387986D-01
4.9537354651168D 00  3.1698904539412D-01  3.9249016659641D-01
5.5729523982564D 00  9.0356307724612D-01  5.1736901733676D-01
6.1921693313960D 00  1.2000000307160D 00 -3.3585912751935D-08

-INTEGRATION STEPS = 2776-
-TRUNCATION ERROR / STEP LESS THAN 1.0D-09
```

```
ASP (CDE-A) PROGRAM OUTPUT      COULOMB FUNCTION

PARAMETERS
EL= 0.0                 ETA= 5.000000E-01
CL= 3.766858E-01

 RHO            W(I), I=1,NE OR W(I)/F(I), I=1,NE
0.0             0.0              3.766858E-01
1.000000E 00  5.166003E-01  5.929234E-01
2.000000E 00  1.021117E 00  3.296018E-01
3.000000E 00  1.043202E 00 -3.169894E-01
4.000000E 00  4.192396E-01 -8.667134E-01
5.000000E 00 -4.904488E-01 -8.331279E-01
6.000000E 00 -1.028593E 00 -1.643893E-01
7.000000E 00 -7.674270E-01  6.531553E-01
8.000000E 00  1.035077E-01  9.621581E-01
9.000000E 00  8.880071E-01  4.885538E-01
1.000000E 01  9.391693E-01 -3.957638E-01

-INTEGRATION STEPS = 122-
-TRUNCATION ERROR / STEP LESS THAN 1.0E-05
```

# APPENDIX C

## JCL AND DECK SETUP

```
  job card
//JOBLIB DD DSN=SYS1.SNOBOL,DISP=SHR
// EXEC ASP
//ASP.INPUT DD *
```

ASP program

```
/*
//GO.DATA5 DD *
```

data cards specified by TABULAR statements
arranged in the order requested

REFERENCES


1. A. Nordsieck, "Numerical Integration of Ordinary Differential Equations," Mathematics of Computation, v. 16, 1962, p.22-49.

2. H. R. Lewis, Jr. and E. J. Stovall, Jr. "Comments on a Floating Point Version of Nordsieck's Scheme for the Numerical Integration of Differential Equations," Mathematics of Computation, v. 21, 1967, p. 157-161.

3. Roland Bulirsch and Josef Stoer, "Numerical Treatment of Ordinary Differential Equations by Extrapolation Methods," Numerische Mathematik, v. 8, 1966, p. 1-13.

4. Milton Abramowitz, "Coulomb Wave Functions," in Handbook of Mathematical Functions, fifth edition, U. S. Dept. of Commerece National Bureau of Standards Applied Mathematics Series 55, 1965, p. 537-554.